UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/941,142	08/28/2001	Jeffrey Meng Wah Chan	P-2628 CNT	7123

24209   7590   11/19/2008
GUNNISON MCKAY & HODGSON, LLP
1900 GARDEN ROAD
SUITE 220
MONTEREY, CA 93940

EXAMINER
HUISMAN, DAVID J

ART UNIT	PAPER NUMBER
2183	

MAIL DATE	DELIVERY MODE
11/19/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

PTOL-90A (Rev. 04/07)

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE _3_ MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1) ☒ Responsive to communication(s) filed on _18 August 2008_.

2a) ☒ This action is **FINAL**.   2b) ☐ This action is non-final.

3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4) ☒ Claim(s) _21-23,25-35,37-47 and 49-58_ is/are pending in the application.

    4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5) ☐ Claim(s) _____ is/are allowed.

6) ☒ Claim(s) _21-23,25-35,37-47 and 49-58_ is/are rejected.

7) ☐ Claim(s) _____ is/are objected to.

8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9) ☒ The specification is objected to by the Examiner.

10) ☒ The drawing(s) filed on _28 August 2001_ is/are: a) ☒ accepted or b) ☐ objected to by the Examiner.

    Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

    Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

    a) ☐ All   b) ☐ Some * c) ☐ None of:

      1. ☐ Certified copies of the priority documents have been received.

      2. ☐ Certified copies of the priority documents have been received in Application No. _____.

      3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

    * See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)

2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.

4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .

5) ☐ Notice of Informal Patent Application

6) ☐ Other: _____.

DETAILED ACTION

1.   Claims 21-23, 25-35, 37-47, and 49-58 have been examined.

Papers Submitted

2.   It is hereby acknowledged that the following papers have been received and placed of

record in the file: Amendment as received on 8/18/2008.

Specification

3.   Due to amendments which further limit the claimed invention, the title of the invention is

no longer descriptive. A new title is required that is clearly indicative of the invention to which

the claims are directed.

Claim Rejections - 35 USC § 103

4.   The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all

obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in
> section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are
> such that the subject matter as a whole would have been obvious at the time the invention was made to a person
> having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the
> manner in which the invention was made.

5.   Claims 42 and 56-58 are rejected under 35 U.S.C. 103(a) as being unpatentable over

Konigsfeld et al., U.S. Patent No. 5,420,991 (herein referred to as Konigsfeld), in view of Witt,

U.S. Patent No. 6,212,622.

6.   Referring to claim 42, Konigsfeld has taught a processor adapted to:

a) speculatively dispatch a load operation to a cache unit and determining, following said

speculative dispatch, whether read-after-write hazards associated with the load operation are

present based on information in a read-after-write field in a load buffer for said load operation.

See the abstract. Note that a load, which occurs after a store, when speculatively executed past

the store, causes a RAW hazard which is ultimately detected by snooping/examining an entry in

the load buffer for that load. Note that the determination as to whether a hazard is present is

based on information in a tag field (column 4, lines 22-30).

b) handle a datum from the cache unit for the speculatively dispatched load operation based, at

least in part, on the determining. See the abstract and column 8, lines 4-16, and note that when a

conflicting store is encountered (hazard occurs), the load has improperly executed, and

consequently, the load is canceled (and the data it loaded is discarded). If a hazard does not

exist, then the data was properly loaded and may be used by a subsequent instruction.

c) Konigsfeld has not taught that each bit in said read-after-write hazard field corresponds to a

different entry in a store buffer and further wherein said each bit has a first state when said

corresponding entry must be executed before said load operation and a second state otherwise.

However, Witt has taught such a concept. See the abstract, Fig.3, column 2, line 36, to column

3, line 7, and column 12, line 32, to column 13, line 35. Specifically, each load instruction has

an associated dependency vector having bit entries which correspond to previous instructions.

When these previous instructions have executed, no hazard exists, and the load may continue.

Otherwise, the dependency vector indicates that these previous instructions have not executed

and a hazard exists, and therefore, the load must wait. One of ordinary skill in the art would

have recognized that both Konigsfeld and Witt are concerned with correct ordering of load/store

operations. Witt's way of handling the ordering allows for flexible scheduling and a wide

variety of dependencies at a high frequency. Hence, it would have been obvious to one of

ordinary skill in the art at the time of the invention to modify Konigsfeld such that the storing of

a store buffer ID is replaced with Witt's concept of including a dependency vector wherein each

bit in the vector has a first state when said corresponding entry must be executed before said load

operation and a second state otherwise.

7.   Referring to claim 56, Konigsfeld has taught a method of locking a resource in a

processor, the method comprising:

a) dispatching for speculative execution a load operation prior to determining whether a read-

after-write hazard exists between the load operation and a store operation with said read-after-

write hazard indicated in a read-after-write hazard field in a load. See the abstract and column 4,

lines 22-30. Note that the determination as to whether a hazard is present is based on

information in a tag field. This tag field comprises bits which make up a store buffer ID of a

store instruction previous to it.

b) locking a resource of the load operation incident with execution of the load operation. See

column 6, line 66, to column 7, line 24, and column 8, lines 4-16. Note that a resource is locked

(while a store is occurring, a load from the same resource cannot occur), where the resource is

related to execution of a load.

c) determining whether the read-after-write hazard exists based on information in said read-after-

write hazard field in said load buffer for said load operation and handling a datum returned for

the load operation based, at least in part, on the determining. See column 4, lines 22-30, and

Fig.5. Also see the abstract and column 8, lines 4-16, and note that when a conflicting store is

encountered (hazard occurs), the load has improperly executed, and consequently, the load is

canceled (and the data it loaded is discarded). If a hazard does not exist, then the data was

properly loaded and may be used by a subsequent instruction.

d) Konigsfeld has not taught that each bit in said read-after-write hazard field corresponds to a

different entry in a store buffer and further wherein said each bit has a first state when said

corresponding entry must be executed before said load operation and a second state otherwise.

However, Witt has taught such a concept. See the abstract, Fig.3, column 2, line 36, to column

3, line 7, and column 12, line 32, to column 13, line 35. Specifically, each load instruction has

an associated dependency vector having bit entries which correspond to previous instructions.

When these previous instructions have executed, no hazard exists, and the load may continue.

Otherwise, the dependency vector indicates that these previous instructions have not executed

and a hazard exists, and therefore, the load must wait. One of ordinary skill in the art would

have recognized that both Konigsfeld and Witt are concerned with correct ordering of load/store

operations. Witt's way of handling the ordering allows for flexible scheduling and a wide

variety of dependencies at a high frequency. Hence, it would have been obvious to one of

ordinary skill in the art at the time of the invention to modify Konigsfeld such that the storing of

a store buffer ID is replaced with Witt's concept of including a dependency vector wherein each

bit in the vector has a first state when said corresponding entry must be executed before said load

operation and a second state otherwise.

8.   Referring to claim 57, Konigsfeld in view of Witt has taught a method as described in

claim 57. Konigsfeld has further taught discarding the datum if it is determined the read-after-

write hazard exists. See the abstract and column 8, lines 4-16.

9.   Referring to claim 58, Konigsfeld in view of Witt has taught a method as described in

claim 56. Konigsfeld has further taught unlocking the resource after the datum is returned. See

column 7, lines 4-7, and note that the resource is unlocked when instruction W2 is performed.

Column 8, lines 4-16, show an example where W2 is performed after R2, which is the instruction

that returns the data (loads it). Consequently, unlocking occurs after returning data.

10.   Claims 21-23, 25-35, 37-47, 49-56, and 58 are rejected under 35 U.S.C. 103(a) as being

unpatentable over Barlow, U.S. Patent Number 5,168,564 (herein referred to as Barlow) in view

of Witt.

11.   Referring to claims 21, 32, and 45 Barlow has taught a method comprising:

a) locking a resource to be accessed by execution of a first instruction in an instruction pipeline.

See Barlow, column 1, lines 50-61, column 2, lines 40-64, and column 5, lines 13-14.

b) determining, after said locking, whether a read-after-write hazard exists between an accessing

portion of the first instruction, when executed in said instruction pipeline, and a portion of a

second instruction based, at least in part, on order of the first instruction with respect to the

second instruction. Again, see Barlow, column 1, lines 50-61, and column 2, lines 40-64, and

note that the locking is performed in order to remedy any hazard associated with a second

instruction's access a resource also being accessed by a first instruction during execution.

c) Barlow has not taught speculative execution of a first instruction. However, the examiner

asserts that speculative execution and its advantages are well known and accepted in the art.

Speculative execution of instructions reduces the cost (in cycle time) of conditional branch

instructions because the processor may predict which way the system will branch (before the

branch is actually resolved), and begin speculative execution along the predicted path. If the processor turns out to be correct, then no time was wasted in resolving the branch. Hence, speculative execution is an optimization technique. As a result, in order to further optimize the processor of Barlow, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Barlow such that the first instruction is speculatively executed.

d) Barlow has further not taught that the order of the first instruction with respect to the second instruction is indicated in a read-after-write hazard field for said first instruction in a buffer of a load/store unit of a processor including said instruction pipeline wherein each bit in said read-after-write hazard field corresponds to a different entry in a store buffer and further wherein said each bit has a first state when said corresponding entry must be executed before said first instruction and a second state otherwise. However, Witt has taught such a concept. See the abstract, Fig.3, column 2, line 36, to column 3, line 7, and column 12, line 32, to column 13, line 35. Specifically, each load instruction has an associated dependency vector having bit entries which correspond to previous instructions. When these previous instructions have executed, no hazard exists, and the load may continue. Otherwise, the dependency vector indicates that these previous instructions have not executed and a hazard exists, and therefore, the load must wait. One of ordinary skill in the art would have recognized that both Konigsfeld and Witt are concerned with correct ordering of load/store operations. Witt's way of handling the ordering allows for flexible scheduling and a wide variety of dependencies at a high frequency. Hence, in order to provide efficient handling of hazards, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Barlow such that the order of the first instruction with respect to the second instruction is indicated in a read-after-write hazard field for

said first instruction in a buffer of a load/store unit of a processor including said instruction pipeline wherein each bit in said read-after-write hazard field corresponds to a different entry in a store buffer and further wherein said each bit has a first state when said corresponding entry must be executed before said first instruction and a second state otherwise.

12.   Referring to claims 22, 34, and 46, Barlow in view of Witt has taught wherein the locking is performed prior to the first instruction entering a trap stage of said instruction pipeline (Barlow column 7 line 60-column 8 line 3, figure 4a, column 5 lines 9-18; the fault, which is the same thing as a trap, or exception, causes the cancel command, but this is after the lock has already occurred). It should be realized that a trap stage could be any point within the processing of the instruction in which a fault is fixed. Clearly, if a resource is already locked, and it needs to be unlocked (column 9, lines 35-36), then the locking is performed before the error is fixed in a "trap stage".

13.   Referring to claims 23, 35 and 47, Barlow in view of Witt has taught wherein the first instruction is an atomic instruction including a portion to lock the resource and a portion to unlock the resource (Barlow column 1 lines 50-61, and column 8, lines 4-6; the resource is locked at the read portion and reset after the write portion of the operation).

14.   Referring to claims 25, 37, and 49, Barlow in view of Witt has taught wherein the locking includes: locking the resource during an effective address calculation stage of said instruction pipeline (Barlow column 4 lines 35-51, column 5 lines 9-19). Clearly, before a resource is locked, its location must be determined.

15.   Referring to claims 26, 38, and 50, Barlow in view of Witt has taught wherein the locking

includes locking at least a portion of a cache (Barlow column 5 lines 26-40, column 9 line 53-

column 10 line 16).

16.   Referring to claims 27, 39, and 51, Barlow in view of Witt has taught wherein the locking

includes locking at least one memory address (Barlow column 5 lines 26-40, column 9 line 53-

column 10 line 16; every entry in the cache is a memory address).

17.   Referring to claims 28, 40, and 52, Barlow in view of Witt has taught further comprising

unlocking the resource no later than a time at which the first instruction exits said instruction

pipeline, regardless of whether the first instruction is cancelled (Barlow column 1 lines 50-61;

the resource is locked at the read portion and reset after the write portion of the operation - after

the write portion of the operation, the process is complete and therefore leave the pipeline).

Clearly, when an instruction leaves the pipeline, all processing corresponding to that instruction

will have been finished. Therefore, if an instruction specifies unlocking, then unlocking will

have to occur before the instruction leaves the pipeline (completes).

18.   Referring to claims 29 and 53, Barlow in view of Witt has taught wherein said unlocking

the resource includes:

        unlocking the resource in the normal course of executing the first instruction (Barlow

column 1 lines 50-61; the resource is locked at the read portion and reset after the write portion

of the operation – after the write portion of the operation, the process is complete and therefore

leave the pipeline).

19.   Referring to claims 30, 41, and 54, Barlow in view of Witt has taught wherein said

unlocking the resource includes preventing a write portion of the first instruction from altering

information held in at least a portion of the resource (Barlow column 2 lines 40-64 – the other

resources are not affected).

20.   Referring to claims 31 and 55, Barlow in view of Witt has taught wherein said preventing

a write portion from altering information includes suppressing writing a value to an architectural

storage location (Barlow column 2 lines 40-64; since the operation is being canceled, there will

be no write-back to the registers).

21.   Referring to claim 33, Barlow in view of Witt has taught further comprising a plurality of

processing cores, wherein respective processing cores are adapted to lock the resource in

response to respective accesses by respective first instructions prior to determining whether said

read-after-write hazard exists between the respective accesses and the second instruction (Barlow

column 9 lines 3-26; multiple cores have access to the same resource).

22.   Referring to claim 42 Barlow has taught a processor adapted to:

a) speculatively dispatch a load operation to a cache and determine, following said speculative

dispatch, whether read-after-write hazards associated with the load operation are present. See

Barlow column 1, lines 50-61, and column 2, lines 40-64. Note that RAW hazards are checked

for because the first instruction is to write before the second instruction reads a location. Since

the lock occurs prior to determination of a hazard, the instruction has experienced "speculative

dispatch" since the instruction speculatively locks the resource.

b) handle a datum from the cache unit for the speculatively dispatched load operation based, at

least in part, on the determining. For a subsequent instruction seeking to access the same data as

the load instruction, the data can be handled in one of two ways. If a hazard exists, then that data

is not made available to the second instruction until the first read-modify-write (RMW)

instruction (which includes the load) is finished with it. If there is no hazard (i.e., the second

instruction does not need to use the data while the first RMW is operating on it), then the data

will be made available to the second instruction. See column 1, lines 50-61.

c) Barlow has not taught determining whether hazards are present based on information in a

read-after-write hazard field in a load buffer for said load operation wherein each bit in said read-

after-write hazard field corresponds to a different entry in a store buffer and further wherein said

each bit has a first state when said corresponding entry must be executed before said load

operation and a second state otherwise. However, Witt has taught such a concept. See the

abstract, Fig.3, column 2, line 36, to column 3, line 7, and column 12, line 32, to column 13, line

35. Specifically, each load instruction has an associated dependency vector having bit entries

which correspond to previous instructions. When these previous instructions have executed, no

hazard exists, and the load may continue. Otherwise, the dependency vector indicates that these

previous instructions have not executed and a hazard exists, and therefore, the load must wait.

One of ordinary skill in the art would have recognized that both Barlow and Witt are concerned

with correct ordering of load/store operations (which is inherent to ensure correct results). Witt's

way of handling the ordering allows for flexible scheduling and a wide variety of dependencies

at a high frequency. Hence, it would have been obvious to one of ordinary skill in the art at the

time of the invention to modify Barlow to include Witt's concept of including a dependency

vector wherein each bit in the vector has a first state when said corresponding entry must be

executed before said load operation and a second state otherwise.

23.   Referring to claim 43, Barlow in view of Witt has taught the processor of claim 42

wherein the processor is adapted to lock a resource associated with the load operation

concurrently with dispatching the load operation (Barlow column 1 lines 50-61, column 2 lines

40-64; the lock indicator, or mechanism can be canceled after it is set once it is determined that

the command using the resource that is locked is invalid, therefore the resource is being locked

before the command has been determined to have hazards, and before the command is known to

go until completion, which goes along with the definition of prior to a determination of a hazard

in the instant application at page 2 line 23-page 3 line 4 – the first portion of a read modify write

is a read from memory, which is the same as a load type instruction – the resource is locked

during the read portion).

24.   Referring to claim 44, Barlow in view of Witt has taught the processor of claim 43

wherein the processor is further adapted to unlock the resource associated with the load operation

no later than a time at which an instruction implementing the load operation exits an instruction

pipeline, regardless of whether the instruction is cancelled before exiting the instruction pipeline

(Barlow column 1 lines 50-61; the resource is locked at the read portion and reset after the write

portion of the operation – after the write portion of the operation, the process is complete and

therefore leave the pipeline).

25.   Referring to claim 56, Barlow has taught a method of locking a resource in a processor,

the method comprising:

a) dispatching for execution a load operation prior to determining whether a read-after-write

hazard exists between the load operation and a store operation indicated in a buffer. See column

1 lines 50-61, column 2 lines 40-64; the lock indicator, or mechanism, can be canceled after

being set once it is determined that the command using the resource that is locked is invalid,

therefore the resource is being locked before the command has been determined to have hazards,

and before the command is known to go until completion, which goes along with the definition

of prior to a determination of a hazard in the instant application at page 2 line 23-page 3 line 4.

In order to detect a hazard between two instructions, a first instruction must be dispatched and

executed. The first RMW instruction includes a load, (i.e., read). The second instruction

includes a store (i.e., a write). So, in order to determine if the second RMW conflicts with the

first, the first must be dispatched. Also, instructions are inherently stored in some buffer before

dispatch.

b) locking a resource of the load operation incident with execution of the load operation. See

column 1, lines 50-61, and column 2, lines 40-64.

c) determining whether the read-after-write hazard exists and handling a datum returned for the

load operation based, at least in part, on the determining. For a subsequent instruction seeking to

access the same data as the load instruction, the data can be handled in one of two ways. If a

hazard exists, then that data is not made available to the second instruction until the first read-

modify-write (RMW) instruction (which includes the load) is finished with it. If there is no

hazard (i.e., the second instruction does not need to use the data while the first RMW is operating

on it), then the data will be made available to the second RMW. See column 1, lines 50-61.

d) Barlow has not taught speculative execution of a load operation. However, the examiner

asserts that speculative execution and its advantages are well known and accepted in the art.

Speculative execution of instructions reduces the cost (in cycle time) of conditional branch

instructions because the processor may predict which way the system will branch (before the

branch is actually resolved), and begin speculative execution along the predicted path. If the

processor turns out to be correct, then no time was wasted in resolving the branch. Hence,

speculative execution is an optimization technique. As a result, in order to further optimize the processor of Barlow, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Barlow such that the load operation is speculatively executed.

e) Barlow has not taught "…indicated in a read-after-write hazard field in a load buffer wherein each bit in said read-after-write hazard field corresponds to a different entry in a store buffer and further wherein said each bit has a first state when said corresponding entry must be executed before said load operation and a second state otherwise. However, Witt has taught such a concept. See the abstract, Fig.3, column 2, line 36, to column 3, line 7, and column 12, line 32, to column 13, line 35. Specifically, each load instruction has an associated dependency vector having bit entries which correspond to previous instructions. When these previous instructions have executed, no hazard exists, and the load may continue. Otherwise, the dependency vector indicates that these previous instructions have not executed and a hazard exists, and therefore, the load must wait. One of ordinary skill in the art would have recognized that both Barlow and Witt are concerned with correct ordering of load/store operations (which is inherent to ensure correct results). Witt's way of handling the ordering allows for flexible scheduling and a wide variety of dependencies at a high frequency. Hence, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Barlow to include Witt's concept of including a dependency vector wherein each bit in the vector has a first state when said corresponding entry must be executed before said load operation and a second state otherwise.

26.   Referring to claim 58, Barlow in view of Witt has taught a method as described in claim 56. Barlow has further taught unlocking the resource after the datum is returned. See column 1, lines 53-57. After the datum is modified and returned to the cache, the lock is reset (unlocked).

Response to Arguments

27.   Applicant's arguments have been considered but are moot in view of the new ground(s)

of rejection.

Conclusion

28.   Applicant's amendment necessitated the new ground(s) of rejection presented in this

Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a).

Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE

MONTHS from the mailing date of this action. In the event a first reply is filed within TWO

MONTHS of the mailing date of this final action and the advisory action is not mailed until after

the end of the THREE-MONTH shortened statutory period, then the shortened statutory period

will expire on the date the advisory action is mailed, and any extension fee pursuant to 37

CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event,

however, will the statutory period for reply expire later than SIX MONTHS from the date of this

final action.

Any inquiry concerning this communication or earlier communications from the

examiner should be directed to DAVID J. HUISMAN whose telephone number is (571)272-

4168. The examiner can normally be reached on Monday-Friday (8:00-4:30).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's

supervisor, Eddie Chan can be reached on (571) 272-4162. The fax phone number for the

organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent

Application Information Retrieval (PAIR) system. Status information for published applications

may be obtained from either Private PAIR or Public PAIR. Status information for unpublished

applications is available through Private PAIR only. For more information about the PAIR

system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR

system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would

like assistance from a USPTO Customer Service Representative or access to the automated

information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/David J. Huisman/
Primary Examiner, Art Unit 2183
October 28, 2008